한국산학기술학회논문지
Vol. 11, No. 6 pp. 2269-2275, 2010

Extended hybrid genetic algorithm for solving Travelling Salesman Problem with sorted population

Yugay Olga[1], Na Hui Seong[1], Lee Tae Kyung[1] and Ko Il Seok[1*]
[1]**Department of Computer and Multimedia, Dongguk University**

Traveling Salesman 문제 해결을 위한 인구 정렬 하이브리드 유전자 알고리즘

유가이올가[1], 나희성[1], 이태경[1], 고일석[1*]
[1]동국대학교 전자계산학부

Abstract The performance of Genetic Algorithms (GA) is affected by various factors such as parameters, genetic operators and strategies. The traditional approach with random initial population is efficient however the whole initial population may contain many infeasible solutions. Thus it would take a long time for GA to produce a good solution. The GA have been modified in various ways to achieve faster convergence and it was particularly recognized by researchers that initial population greatly affects the performance of GA. This study proposes modified GA with sorted initial population and applies it to solving Travelling Salesman Problem (TSP). Normally, the bigger the initial the population is the more computationally expensive the calculation becomes with each generation. New approach allows reducing the size of the initial problem and thus achieve faster convergence. The proposed approach is tested on a simulator built using object-oriented approach and the test results prove the validity of the proposed method.

요 약 유전자 알고리즘은 매개변수와 유전자 연산자 그리고 계획과 같은 다양한 요인들에 의해 영향을 받으며, 전통적인 방법을 통한 문제의 해결은 효율적이지만 전체적으로는 실행 가능성의 문제와 결과의 도출에 걸리는 시간의 문제가 있을 수 있다. 이에 따라 전통적인 유전자 알고리즘은 다양한 방법으로 수정 및 적용되어 질 수 있다. 본 연구는 Travelling Salesman 문제를 해결하기 위해 초기에 정렬된 인자를 사용하여 수정된 유전자 알고리즘을 적용하였다. 본 연구를 통한 접근 방법은 초기 문제의 크기를 줄이며 또한 빠른 복합 수렴을 달성하였다. 또한 제안된 방법은 객체 지향 접근을 사용한 시뮬레이터를 통해 테스트 되었고 그 결과는 제안된 방법의 타당성을 입증하였다.

Key Words : Traveling Salesman Problem, Genetic Algorithm

1. Introduction

Genetic Algorithms (GA) by Goldberg are based on the principle of natural selection[1]. The GA consists of several components and its performance is consequently effected by the performance and quality of each of the components. Thus such factors as genetic operators, parameters and strategies [2] influence the GA performance. There have been numerous adaptations of the GA that modify those factors and thus resulting in various Hybrid GA (HGA)[2-5].

Traveling Salesman Problem (TSP) is a classic search problem known to be NP-complete. Solving TSP using Genetic Algorithm is widely researched topic and there is a number of HGA applied to TSP.

This study proposes to modify generation of initial population. In fact, it was recognized by Togan et al. [2] et al. that initial population has great affect on the

*교Corresponding Author : Ko Il Seok(isko@dongguk.edu)
Received February 23, 2010 Revised April 29, 2010 Accepted June 18, 2010

performance and the ability of the GA because the consequent generations in GA will heavily depend on it.

Traditional approach with random initial population can be efficient; however the whole initial population may contain great number of infeasible solutions. And it would take many generations for it to evolve into a feasible solution. Some studies modified strategies for generating initial population; for instance, improved GA with initial population strategy and self-adaptive member grouping by Togan et al. [2].

The proposed method main difference lies in its simplicity and efficiency. We assume that a better solution can be obtained faster since initial population is significantly reduced with higher fitness population preserved. In this paper we provide a brief overview of GA, TSP and Hybrid GA. This is followed by the thorough explanation of the proposed method, which is later tested on the simulator. We also present the results of the testing as a proof of the proposed method efficiency.

2. Traveling Salesman Problem with genetic approach

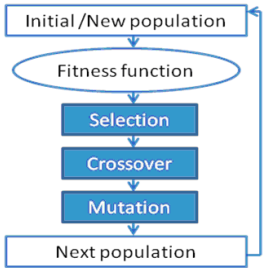
2.1 Traveling Salesman Problem

Traveling Salesman problem (TSP) can be defined as: in given an undirected graph of a number of cities, the objective is to find a path of shortest length (or of minimum cost). The traveling salesman starts at one node and visits the rest of the cities only one time each and finally returns to a starting point. [5]

Let n represent the number of cities, and c_1, c_2, ...c_n represent the cities themselves and d (c_i, c_j) – distinct pair of cities. The goal is to find such a route, or in other words the ordering of cities so that it would minimize the length of the tour . In our case we assume Euclidian TSP. As it is well known, TSP is NP-hard optimization problem.

2.2 Genetic Algorithms

Genetic Algorithm (GA) is based on the principles and mechanisms of natural selection and the survival of the fittest concept. This algorithm works well on mixed (continuous and discrete) combinatorial problems, because they are less likely to get 'stuck' in local optima than gradient search methods. Its' disadvantage is being computationally expensive [1,6,7,12,13]. The original GA cycle is depicted in Figure 1.



[Fig. 1] GA cycle adopted from [5]

To understand genetic algorithms it is important to know the following terminology:

Chromosome is a structure defining a specific solution to a problem, consisting of genes. In case of TSP chromosome is a tour.

Genes represents a specific part of a solution, in case of TSP, it is city, or distance between 2 cities

Allele is value assigned to a gene. In case of TSP, it is coordinates or distance between 2 cities.

Population is a set of chromosomes that represent different solutions to the same problem. In case of TSP, it is a set of tours.

Generation is the population of chromosomes that exists at time t. The next generation of this population is a set of chromosomes that exists at time t+1 and the ancestors of the population would be any population that appeared before time t.

Fitness of an individual indicates the "health" of the individuals. Those that are more fit will have a higher fitness level and are more likely to survive and reproduce[1]. In TSP case - fitness is the length of the tour.

There is also a notion of genetic operator, which help to transform population p at time t into population at time t+1. Genetic operators include *selection*, *crossover*, and *mutation*. The selection operator chooses two members (chromosomes) of the present generation in order to participate in the next operations: crossover and mutation, based on their fitness. The crossover operator combines

genes of 2 chosen parents to generate a child. The mutation operator occurs after crossover in the new generation to mutate [1,6,7]. Points to mutate get randomly chosen based on the mutation rate specified by the user. Some researchers suggest including elitism which refers to an act of including best chromosome from previous population, because there is a big probability of losing best chromosome after implementing crossover and mutation operators. [7,8]

2.3 Hybrid GA approaches to solve TSP

The hybrid approaches to Genetic Algorithms involve combination of GA and some local search algorithm, usually stochastic hill climbing, simulated annealing and others. Example can be Hybrid Mutation Genetic Algorithm by Katayama et al. [4] where he suggests a new crossover operation (complete subtour exchange crossover) and stochastic hill climbing as mutation operator. This approach proved to show very fast convergence in comparison with GA with ordinary mutation process.

The other approach to modify GA algorithm is trying to modify the procedures at any of GA phases, such as generation of initial population, selection, mutation. Togan et al [2] suggested improved genetic algorithm with initial population strategy and self-adaptive member grouping. The strategy proposes to start search with specific individuals instead of generating the initial population randomly and allows reaching the global optimum with less number of iterations.

Katayama et al [4]. used a hybrid approach through modifying the mutation phase. Javadi et al. [3] modified genetic algorithm by introducing Neural Network that improved the convergence of the genetic algorithm in search for global optimum.

Marco Dorigo described a method of heuristically generating "good solutions" for TSP by applying an ant colony simulation called ACS. [11] It is based on behavior of real ants to find short paths between food sources and their nest resulting from each ant's preference to follow trail pheromones left by other ants. The shorter the path the more preferred and travelled would be the path.

All of the methods justified and proved its efficiency, however the approach proposed in all methods is complicated and it is believed to take longer calculation time due to its complexity. We believe that proposed technique is simple and straightforward thus yielding better results in shorter time.

3. Proposed Hybrid Approach with Sorted Population (HASP)

The main motivation for proposed method was the underlying theory of GA, that better parents would produce better offspring. In traditional approach the initially generated population normally contains various chromosomes, including not “fit” chromosomes. Such bad chromosomes may be a cause of producing less healthy generation, which means it would take it longer time to converge and produce feasible result. This allows us to suggest modification to traditional GA by applying sorting and seeding to the initial population. In particular, the modification involves generating a large initial pool of population, then sorting it and deleting a certain percentage of population that has not fit “chromosomes”, in case of TSP – large total distance.

We assume that with very large initial population there is a high probability of having good chromosomes in population. The second step – deleting “bad” chromosomes, will ensure that more fit population is retained for further generations. That has a two-fold benefit: firstly, a fitter initial population will result in fitter generations; secondly, the size of subsequent population will be significantly reduced since “bad” chromosomes are deleted from initial population and thus it will take less time for modified GA to converge.

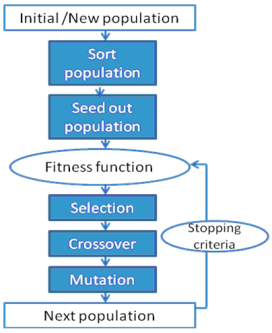
There are a number of well known sorting algorithms. The proposed strategy uses merge sort for sorting the population due to its stability and $O(n \log n)$ time complexity in average and in the worst case[13].

The procedure of new hybrid GA is illustrated in the Figure 2. And updated GA cycle can be seen from figure 3.

```
Procedure   hybrid GA
01: begin
02: k:=0;
03: generate very large initial population of feasible
    solutions P(k)
```

```
04: sort the P(k)
05: Delete  a given percentage of solutions
    in P(k)  that have low fitness
06: while stopping-criterion!=yes do
07:   roulette selection of parents
08:   crossover
09:   mutation
10: end
11: return best solution
```

[Fig. 2] Proposed hybrid GA algorithm with sorted population



[Fig. 3] Hybrid GA cycle with initial population seeding

Proposed hybrid GA algorithm description (Refer to figure 3):

Step1: generate large initial population (randomly) – the size of population will depend on the number of genes.

Step2: sort the population

Step3: seed the population – delete not "fit" chromosomes, it can be a certain percentage of the population.

Step4: apply fitness function

Step5: based on the fitness function perform selection of the suitable chromosome. There are various selection options, e.g. roulette selection, rank selection, or steady state selection. For our experiment we chose roulette selection, since chromosomes with higher fitness will have larger chance of being selected for the next step crossover.

Step6: Crossover is performed on this stage, such that genes of two selected chromosomes (parents) are interchanged to produce a new chromosome (offspring), which is subsequently added to the next population. There are also a number of different crossover [5,14]. In our experiment we used single point crossover. It is based on permutation encoding a single crossover point is selected, till this point alleles from parent 1 are copied into the offspring, then the second parent is scanned and if the allele is not yet in the offspring it is added to offspring.

Step7: Mutation is an optional step. It is desirable because occasional mutation would ensure that GA will not result in premature convergence. Mutation is performed only on some chromosomes randomly selected from newly generated population. Mutation involves random interchange in the order of genes of the chromosome. After change the chromosome in reinserted into the population.

Step8: Check against the stopping criteria. The stopping criteria can be the number of generations (cycles the algorithm is executed), or desirable distance. Go to step4. This cycle repeats until the stopping criteria is met.

Criticism

One may criticize that new approach focuses on "strictly fit" population leading to a premature convergence. It can be argued that our approach does not have a strict focus on that but rather merely gets rid of potentially bad population that would only stagger the development of better population. Once again, we stress that algorithms heavily relies on the GA main concept – natural selection.

4. Experiment

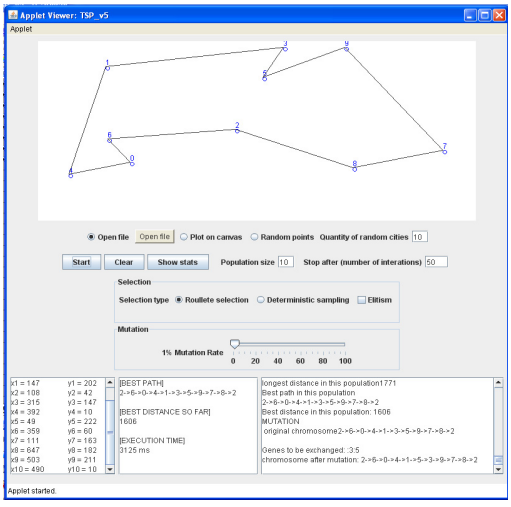
The experiment was carried out by building applet simulator on Java language using Object-Oriented Programming.

Assumptions: Undirected complete graph is assumed for TSP. The experiment was carried out with the following settings: population – varied from 10-50 (10/15/20/25/30/35/40/45/50), crossover - single point crossover, selection – roulette selection, mutation rate –

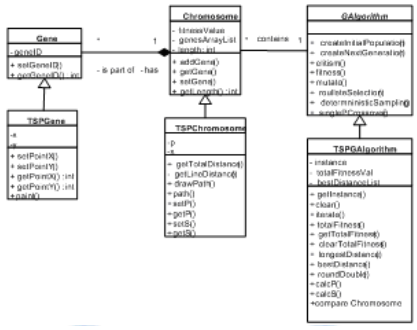
1% in order to ensure that solution will not stuck in local optima, total previous population replacement with new population, no elitism, the number of iterations is deliberately set to a high number, because the simulator is built so that it automatically quits after it finds local optima.

Testing environment: Intel (R), Core (TM)2 CPU T5600 @ 1.83GHz RAM 1.00 GB

Simulator snapshot can be seen in Figure 4.



[Fig. 4] Simulator snapshot



[Fig. 5] Conceptual model

4.1 Design

Object oriented approach to building the simulator is reflected in Conceptual UML CLASS diagram is presented in Figure 5. Since Genetic Algorithms have wide range of implementation abstraction has been used and it is reflected in class diagram. For example, TSPGene is Gene, however it has TSP specific attributes and methods like x, y coordinates and etc. Similar case is with Chromosome, GAlgorithm.

Superclasses define the general for all problems variables and methods. The subclasses define specific methods, for example subclass TSPGAlgorithm defines such TSP specific methods as longestDistance(), bestDistance() and etc.

Encoding. Due to TSP specific nature, the permutation encoding is used for it. However, this encoding should be dealt carefully with

when used in crossover. For example, tours like:

1) ABCD
2) BCDA
3) CDAB
4) DABC

in fact are the same. Though the starting city is different in every case, the course of travel includes all the cities in the same order of traveling, which make the tours completely identical. This has been dealt with in our simulator, by implementing a function compareChoromosome, which compares the chromosomes against each other. This function allows avoiding using identical tours for crossover and thus avoiding unnecessary calculations.

Sorting.

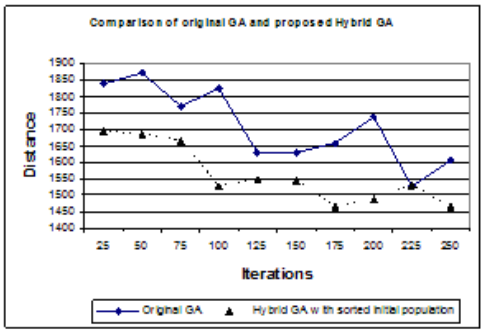
We used existing sorting algorithms available in java Collections class library for sorting the initial population. This algorithm produces results in O(logN) in the worst case and proved to be efficient enough for the simulation.

4.2 Results

The results of testing indicate that algorithm described in this paper yielded better solutions in all trials (faster) and what is more important in a smaller number of generations that original GA. The results were averaged for 10 trials and presented in the Figure 4.

It can be seen though the numbers fluctuate in the result, overall the performance of the proposed Hybrid GA shows better result.

The original GA was also implemented by us in order to compare its results with the Hybrid GA. In the test we deleted 50% of initial population. For 20 initial points we used population of 100 for the Hybrid GA with sorted population and population of 50 for original GA in order to make the results comparable.



[Fig. 6] Comparison of original GA and proposed Hybrid GA

5. Conclusion and future work

In this paper we introduced simple and yet efficient new Hybrid GA with sorted initial population. We were motivated by the GA basic theory that better population is achieved with better previous population. We have presented overview of other Hybrid GA and proved that our method has faster convergence and yields same/better results then other HGA. The testing was carried out on the object-oriented built simulator and the results proved the validity of the method.

Future scope of work on the proposed algorithm is quite large. Study can be conducted to analyze the performance of algorithm with different percentage of initial population being deleted. Also, the study of how large the size of initial population should be to create a more efficient and effective flow of the algorithm.

References

[1] D.E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning, Addison”, Wesley, pp. l-88, 1989.

[2] Togan V., Daloglu A. “An Improved genetic algorithm with initial population strategy and self-adaptive member grouping”, Computers and Structures, Volume 86, Issue 11-12, pp. 1204-1218, 2008.

[3] Javadi A.A., Farmani R., T.P. Tan “A Hybrid intelligent genetic algorithm”, Advanced Engineering Informatics, Volume 19, Issue 4, pp. 255-262, 2005.

[4] K. Katayama, H. Sakamoto, "The Efficiency of Hybrid Mutation Genetic Algorithm for the Travelling Salesman Problem", Mathematical and Computer Modelling, Volume 31, pp. 197-20, 2000.

[5] Buthainah Fahran Al-Dulaimi, and Hamza, A. Ali “Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique (TSPGA) Proceedings of World Academy of Science, Engineering and Technology, Volume 28, pp. 296-302, 2008.

[6] G.J.E. Rawlins, “Foundations of Genetic Algorithms”, Morgan Kaufmann Publishers, TSPBIB, TSP library, 1991.
http://www.ing.unlp.edu.ar/cetad/mos/TSPBIB_home.htm

[7] L. Chambers, “Practical Handbook of Genetic Algorithms Applications”, CRC Press, Volume 1, pp, 143-172, 1995.

[8] Liangsheng Qu, Ruixiang Sun, "A synergetic approach to genetic algorithms for solving traveling salesman problem", Information Sciences 117, 267±283, 1999.

[9] TSP library http://www.tsp.gatech.edu/index.html

[10] Lau Tung Leng, “Guided Genetic Algorithm”, University of Essex, A thesis submitted for the degree of Ph.D in Computer Science, Department of Computer Science.

[11] Marco Dorigo, “Ant Colonies for the Traveling Salesman Problem”, IRIDIA, Université Libre de Bruxelles. IEEE Transactions on Evolutionary Computation, 1(1):53–66, 1997.

[12] Lee Sang-Cheol, Yu Jeong-Cheol, “Improved VRP & GA-TSP Model for Multi-Logistics Center”, Journal of the Korea Academia-Industrial cooperation Society, Vol8, Num5, 2007.

[13] Kim Ki-Bong, "Search Method for Consensus Pattern of Transcription Factor Binding Sites in Promoter Region", Journal of the Korea Academia-Industrial cooperation Society, Vol9, Num5, 2008.

유가이올가(Yugay Olga) **[준회원]**

- 2007년 6월 : 웨스트민스터대학교 비즈니스 컴퓨팅학과 (학사)
- 2010년 2월 : 동국대학교 전자계산학과 (석사)

<관심분야>
인공 지능, 멀티 - 에이전트 시스템, 유전자 알고리즘

나 희 성(Na Hui Seong) **[준회원]**

- 2006년 2월 : 충북대학교 컴퓨터공학과 졸업(학사)
- 2008년 9월 ~ : 동국대학교 전자계산학과 (석사)

<관심분야>
웹 캐싱, 데이터 마이닝, 멀티미디어 디지털 콘텐트

이 태 경(Lee Tae Kyung) **[정회원]**

- 1980년 2월 : 동국대학교 전자계산학과 학사
- 1987년 8월 : 숭실대학교 전자계산학과 석사
- 1994년 8월 : 숭실대학교 전자계산학과 공학 박사
- 현재 : 동국대학교 컴퓨터학과 교수

<관심분야>
지식공학, 에이전트 시스템, Bio informatics, ICAI

고 일 석(Ko Il Seok) **[정회원]**

- 1989년 2월 : 경북대학교 컴퓨터학부 학사
- 1996년 2월 : 경북대학교 컴퓨터학부 석사
- 2006년 2월 : 연세대학교 컴퓨터학부 공학 박사
- 현재 : 동국대학교 컴퓨터학과 교수

<관심분야>
네트워크 기반의 미디어 솔루션/서비스, 유/무선 통신망에서 안전한 멀티미디어 디지털 콘텐트 전송시스템, 사용자 특성 기반의 웹시스템/서비스, 지능형 보안시스템